ECE 4750 Computer Architecture
Example Pipeline Diagram

revision: 2016-09-23-14-03

This is not a quiz, but it does give you another chance to draw a pipeline diagram.

Problem 1. Resolving Data Hazards with Stalling

In this problem, you will identify architectural data dependencies, and then illustrate how we can resolve potential data hazards through stalling.

Part 1.A Architectural Data Dependencies

Consider the following sequence of TinyRV1 instructions. Draw arrows to indicate the architectural RAW dependencies between instructions.

```
addi   x1,    x2,    1

lw     x3,    0(x1)

lw     x4,    0(x3)

add    x5,    x3,    x1
```

Part 1.B Pipeline Diagram Illustrating Stalling

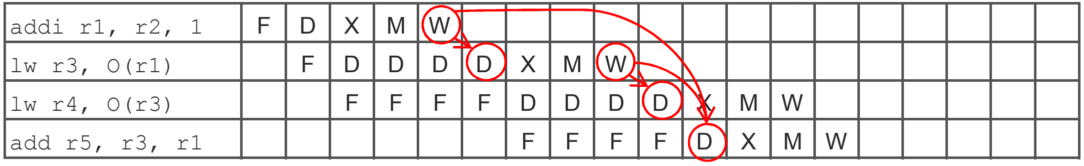
Draw a pipeline diagram that illustrates how these four instructions would execute on a pipelined TinyRV1 processor that uses only stalling to resolve data hazards. You should assume the processor includes absolutely no bypassing. Add arrows to your pipeline diagram to indicate all microarchitectural RAW dependencies (those that result in passing values through registers and those that result in using the bypass network). Ensure that your diagram does not have any data hazards.

`addi x1, x2, 1`																			
`lw   x3, 0(x1)`																			
`lw   x4, 0(x3)`																			
`add  x5, x3, x1`																			

ECE 4750 Computer Architecture
Common Mistakes in Pipeline Diagrams

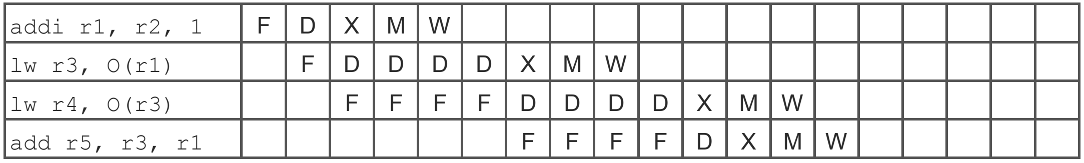
Jason Setter

0. correct answer : correct stalling and all the dependency arrows (No bypass)



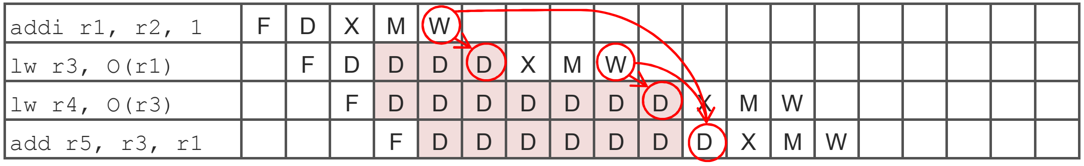
addi r1, r2, 1	F	D	X	M	W														
lw r3, 0(r1)		F	D	D	D	D	X	M	W										
lw r4, 0(r3)			F	F	F	F	D	D	D	D	X	M	W						
add r5, r3, r1							F	F	F	F	D	X	M	W					

1. no arrows : Do not forget the dependency arrows!



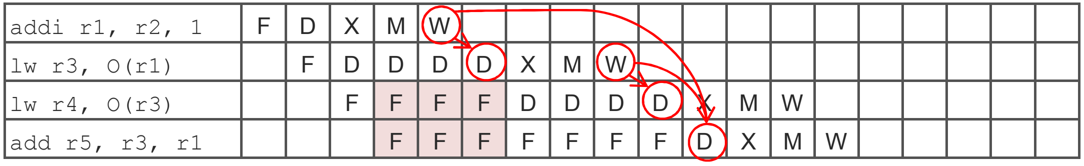
addi r1, r2, 1	F	D	X	M	W														
lw r3, 0(r1)		F	D	D	D	D	X	M	W										
lw r4, 0(r3)			F	F	F	F	D	D	D	D	X	M	W						
add r5, r3, r1							F	F	F	F	D	X	M	W					

2. multiple instructions in D stage : Structure hazard!



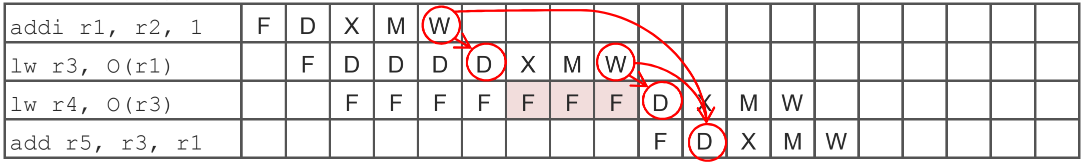
addi r1, r2, 1	F	D	X	M	W														
lw r3, 0(r1)		F	D	D	D	D	X	M	W										
lw r4, 0(r3)			F	D	D	D	D	D	D	D	X	M	W						
add r5, r3, r1				F	D	D	D	D	D	D	D	X	M	W					

3. multiple instructions in F stage : Also, structure hazard!



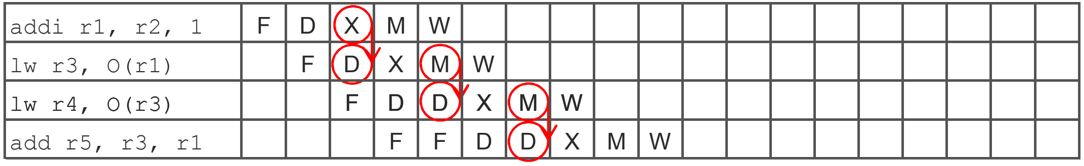
addi r1, r2, 1	F	D	X	M	W														
lw r3, 0(r1)		F	D	D	D	D	X	M	W										
lw r4, 0(r3)			F	F	F	F	D	D	D	D	X	M	W						
add r5, r3, r1				F	F	F	F	F	F	F	D	X	M	W					

4. keep stalling F stage instead of D stage : No need to stall F stage after D stage stops stalling.



addi r1, r2, 1	F	D	X	M	W														
lw r3, 0(r1)		F	D	D	D	D	X	M	W										
lw r4, 0(r3)			F	F	F	F	F	F	F	D	X	M	W						
add r5, r3, r1										F	D	X	M	W					

5. bypassing : Read the problem carefully. It includes absoultely no bypassing.



addi r1, r2, 1	F	D	X	M	W														
lw r3, 0(r1)		F	D	X	M	W													
lw r4, 0(r3)			F	D	D	X	M	W											
add r5, r3, r1				F	F	D	D	X	M	W									

6. wrong bypassing : bypassing should be combinational, even though bypassing is allowed.

`addi r1, r2, 1`	F	D	X	M	W														
`lw r3, 0(r1)`		F	D	D	X	M	W												
`lw r4, 0(r3)`			F	F	F	F	D	X	M	W									
`add r5, r3, r1`							F	F	F	D	X	M	W						

7. getting value from X stage : We need to get the value from D not X stage. Need one more stall.

`addi r1, r2, 1`	F	D	X	M	W														
`lw r3, 0(r1)`		F	D	D	D	X	M	W											
`lw r4, 0(r3)`			F	F	F	D	D	D	X	M	W								
`add r5, r3, r1`						F	F	F	D	X	M	W							

8. getting value from D stage, but combinatinally : No bypass in this problem!

`addi r1, r2, 1`	F	D	X	M	W														
`lw r3, 0(r1)`		F	D	D	D	X	M	W											
`lw r4, 0(r3)`			F	F	F	D	D	D	X	M	W								
`add r5, r3, r1`						F	F	F	D	X	M	W							

9. different name : It's not a deal breaker, but please follow our syntax in this course.

`addi r1, r2, 1`	F	M	X	C	S														
`lw r3, 0(r1)`		R	D	D	D	D	E	M	W										
`lw r4, 0(r3)`			ID	ID	ID	ID	D	D	D	D	Ex	M	Wb						
`add r5, r3, r1`							F	F	F	F	D	R	M	W					